Marooned on Mars: Mind-spinning books for software engineers

by William J. Clancey

Chief Scientist, Human-Centered Computing
NASA/Ames Research Center
Moffett Field, CA 94035 (*)

(*) On leave from the Institute for Human-Machine Cognition, University of West Florida, Pensacola.

To appear in the Automated Software Engineering Journal, published by Kluwer Academic Publishers, in Volume 7, year 2000

I've just arrived on Mars, with 500 days before the next return shuttle. Fortunately, we have email and internet access to Earth (the line is fast, but there's a twenty minute delay on average, almost like time-sharing in the 70s). My powerbook fits fine on my lap, but slouching on the habitat's long couch, I prefer holding a book in my hands, settling down with gleeful anticipation with a warm drink nearby (it's -70c today). So I brought along an armload of books, some for reference, some to read and study again, and others to share with the next generation, as we build our colony. All are mind-spinning, just what we need for opening a new world with new ways of thinking.

To start, I brought along Burrough's (1998) *Dragonfly: NASA and the Crisis Aboard MIR* (New York: HarperCollins Publishers), the story of the Russian-American misadventures on MIR. An expose with almost embarrassing detail about the inner-workings of Johnson Space Center in Houston, this book is best read with the JSC organization chart in hand. Here's the real world of engineering and life in extreme environments. It makes most other accounts of "requirements analysis" appear glib and simplistic. The book vividly portrays the sometimes harrowing experiences of the American astronauts in the web of Russian interpersonal relations and literally in the web of MIR's wiring. Burrough's exposition reveals how handling bureaucratic procedures and bulky facilities is as much a matter of moxie and goodwill as technical capability. Lessons from MIR showed NASA that getting to Mars required a different view of knowledge and improvisation—long-duration missions are not at

all like the scripted and pre-engineered flights of Apollo or the Space Shuttle. Thanks to the efforts of the Human-Centered Computing group at NASA/Ames, the days when engineers separated power, dials, and ethernet ports on opposite sides of the Space Station are past. Why, for our nine month voyage to Mars they even designed the kitchen table to stay open all day! (On the Space Station we had to stow it after every meal to get access to storage below.)

Dragonfly shows the crazy antics of real-world operations; what's the theoretical foundation for improving the design of complex systems? Here I'm well-supplied. Though heavy (using half of my allotted bookcase), I brought along Shapiro's (1992) multiple-volume *Encyclopedia of Artificial Intelligence*. (New York: John Wiley and Sons.) The technical quality of this reference is unsurpassed. Whether you're looking for details about Hidden Markov Models or hermeneutics, it's here, with clarity, accuracy, and good citations. Although well-versed on many of the topics, I find myself turning to this encyclopedia for historical and technical details. With this book for reference, we've created "intelligent" operations assistants in the Mars habitat—built with agents and reusable inference engines—a far cry from the monolithic computer system at Houston's Mission Control, which was ported and changed piecemeal for 30 years (in the name of safety).

Speaking of learning from the past, I've also brought the *ACM Turing Award Lectures: The First Twenty Years 1966-1985* (New York: The ACM Press). I came upon this volume when a friend mentioned Hoare's lecture, "The Emperor's Old Clothes." Hoare shares his stark experience: "The entire Elliott 503 Mark II software project had to be abandoned...equivalent to one man's active working life, and I was responsible, both as designer and manager, for wasting it" (p. 150). Building on this experience, Hoare fervently appeals to us not to allow ADA into the real world. Fortunately for me, NASA was never much for programming fads, so we're not flying ADA on Mars. Anyway, after I searched for Hoare's lecture on the web, I found a collection of other Turing Award lectures and decided I wanted to read and study them all. (I hope we'll soon get the next volume.)

Now although I'm an accomplished programmer (I'm using Visual Basic for Applications to link the astrogeologists' datasheets and reports), my professional role for the Mars base has gravitated to the philosophy of

engineering design. So the bulk of my Martian collection is more about design as a creative process. Here's the core collection:

Alexander, C., et al. (1977). *A pattern language.* New York: Oxford University Press.

Bamberger, J. (1991). *The mind behind the musical ear.* Cambridge, MA: Harvard University Press.

Schön, D. A. (1987). *Educating the reflective practitioner.* San Francisco: Jossey-Bass Publishers.

Wilden, A. (1987). *The rules are no game.* New York: Routledge and Kegan Raul.

These books are about the relation of artifacts, patterns, descriptions, notations, designs, and the process of invention. Paraphrasing Schön, the topic is not how to build, but *what to build.* Each teaches us about the relation of the mind—how ideas form and are related—and humanity's constructed environment. They fundamentally help us understand the relation of individual thoughts and social contexts and how change occurs on various scales—whether an incremental edit to a diagram, a reperceived and reinterpreted rearrangement of parts, or a reciprocated move in a social venture.

Alexander's book is about architectural patterns; yet really it shows how grammar descriptions (or policies) do not strictly generate the world of human artifacts and behavior, but serve as a kind of guide or map. Alexander's ideas are as valuable for designing work places and software tools, as courtyards and bedrooms. Ultimately, Alexander's claim is epistemological: The knowledge of good design is embodied in artifacts developed and improved *in situ* and can never be reduced to written principles or laws. There are two reasons for this: First, we can always reinterpret past successes in new contexts to articulate new heuristics, and second, the thoughts and actions that produced the original artifacts were not themselves bounded by descriptive rules or plans.

Unfortunately, a subgroup of software engineering has taken up the pattern language hammer by the wrong end of the handle: Instead of viewing the pattern perspective as an analytic, requirements analysis and

evaluation technique (for explicating the context in which a tool must operate), they have reduced the idea to more descriptions that should be stored in a computer and used to generate I/O behavior. This can lead to precisely the wrong result—constraining work processes to follow a designer's predescribed workflow regulations. Instead we should view the patterns as improvised, situated arrangements (of facilities, deeds, and materials) and inquire about the aspects of workplace design that enable these patterns to develop . To apply Alexander properly, we need to view workplace patterns as rules of thumb that reflect locally grown relationships in tools and practices; nevertheless, as descriptions they are of broad value for inspiring future work system design. For example, could we develop a pattern language for computer network design in different kinds of collaborative workplaces, such that the language articulates dimensions of size, risk, routine, and skills being employed? (See Clancey, 1995b, 1997b.)

Developed in the work of Bamberger and Schön, Alexander's ideas about emergent organization are manifest as the "situated cognition" theory of knowledge and behavior. The upshot is that within every human action, there is a non-descriptive component, that is, a physical aspect that is not modeled and planned, but a neural level of coordination that reactivates and adapts perceptual categories, concepts, and motor actions. Now, that's a mouthful to be sure, but Bamberger and Schön unfold these ideas carefully with a series of simple examples from music and art, showing the learner's perspective (and how this contrasts with the teacher's terminology and curriculum). (Bamberger and Schön don't really get to the neural memory level—you'll have to read my 1997 book to see the relation.)

Dewey (1896) said all this long ago, but change is slow on Earth. In important ways, computer science is partly responsible for the retrograde epistemology of the 1960s and 70s. The metaphor (and success) of the von Neumann architecture reified and reinforced ideas about memory, knowledge, and learning that—as psychological explanations—were outdated a century ago. The brain works in a different way than today's conventional computers. The brain doesn't execute programs in a literal way, but reactivates perceptual-motor circuits "in line" and generalizes them at the same time. The implications are profound for software engineers (and especially AI specialists like myself). We cannot identify how our tools work with people (e.g., "expert systems" "knowledge

bases"), and if we want to make progress in developing computers with human intelligence, we need a different memory-coordination mechanism (Clancey, 1995a, 1999a). For example, the simplest interpretation of "knowledge management"—a trendy notion in business software today—is based on capturing, storing, and disseminating knowledge. But this equates knowledge with databases and models—a mistake that Dewey said was like confusing a carpenter with his tools.

Perhaps now you can grasp Wilden's title, "The rules are no game." By one interpretation, the game—how people perceive and conceive of their actions—is different from the rules—written procedures of how to behave. The map is not the territory. This ultimately only makes sense from a human cognition standpoint when you realize how wrong the storage metaphor of knowledge is. What's neat about Wilden's book is that he shows how these metaphors have played out in cinematic and social-political settings. Perhaps the antagonism of conservative to liberal political parties has its origins in hierarchical neural processes by which conceptual systems develop: *Assimilation* (highlighting of general values) and *differentiation* (highlighting of diversity). These ways of relating ideas may develop in individuals as mental styles and thus different strategies for reconciling social problems. Such a philosophical analysis only makes sense when you realize that concepts in the human brain are not networks of word definitions, so again "knowledge" and "reasoning" involve real-time adaptive capabilities in people that the present-day computer architectures do not replicate. In particular, the idea of user models has been hampered by shortcomings in the theory of how individuals differ cognitively. On the one hand, non-verbal aspects of cognition are not adequately related to perception and language; and on the other hand, differences of knowledge are reduced to variations in descriptive models. Wilden's work is challenging because he starts with people and real-world experience, rather than tidy theories.

Indeed, enough theory! Back to building a Martian colony. I brought along two books to round out my collection. The first, *Design at Work: Cooperative Design of Computer Systems* (edited by Greenbaum, J., & Kyng, M., Hillsdale, NJ: Lawrence Erlbaum Associates, 1991) is an experimental handbook, telling the story of a group of social and cognitive scientists who put their (sometimes rhetoric-heavy) ideas to practice. My favorite chapter is Wynn's "Taking practice seriously." I smile every time I visualize her account of workflow diagrams, "There are

people helping this block to be what it needs to be—to name it, put it under a heading where it will be seen as a recognizable variant, deciding whether to leave it in or take it out, whom to convey it to" (pp. 56–57). When you impose programmatic processes on people, you might make a mess of the workplace. Bannon's "From social factors to social actors" is equally provocative and mind-spinning. In fact it's all here, from interface design to video interaction analysis, to scenario-based prototyping. Other books tell the story and other projects do it better, but as a primer on how to design software so it fits human purposes, this is the one I brought to reread and exploit.

Actually, with this collection so far, I've come full circle: These are the books that inspired our design of the Mars Arctic Research Station, an analog experiment on Devon Island in the Canadian Arctic (M.A.R.S., 1999). In this extreme setting, we studied scientists and engineers investigating a Mars-like impact crater. We established a baseline for their practices; then following the principles of design in the context of use and *in situ* evaluation, we prototyped tools that would facilitate life and work on Mars (Clancey, in press). For instance, what tools are required to log and analyze rock samples in the field, and indeed to write journal papers before returning home? We showed that the Internet, which provided direct access to colleagues and the public, radically changed the role of "Mission Control" back on Earth.

My last book, Petroski's (1985) "To engineer is human: The role of failure in successful design" (New York: St. Martin's Press) has already been mentioned by two fellow Desert Island readers (Dobson, 1996; Ryan, 1996). But in developing software for space exploration and dealing with the many disasters that will face us on Mars, I want every colonist to read this book. Basically, the book is about *perspective*, and provides a hopeful way of coping with inevitable setbacks. Somehow our society has developed an aversion to failure, making it an indicator of incapability, rather than a stepping stone. For example, NASA's unofficial calculations predict that of an initial crew of six going to Mars, only five will survive to return. Society needs to be ready for that outcome. Somehow it's not enough to remind people of the countless ships and men who were lost in exploration just a few score years ago in searching for the Northwest Passage (Lopez, 1986). Somehow we have the idea that things are different now; we have perfected our methods, so mechanical parts and software never fails. But then when it does, of course, we knock

ourselves down, and think less of our society. Maybe this stems from a lack of external threat that would rally us to new efforts. Indeed, that's one reason why I jumped at the chance to go to Mars. Only really difficult challenges, where losses are inevitable, will reveal how limited our capabilities really are. Whether it's building rockets or robots, we have just begun.

It's a scary place here on Mars. One false move and I'm dead. A hundred steps from the base, I'm in a cold, empty world. All our designs, our automated systems, and our social ideas are young and forming, tenuous, yet growing. Most of the prevalent theory about knowledge and memory on which we build software tools for people is primitive and misleading. And our computer architectures are just making the first steps to self-organizing, "in place" circuitry. That's the critical and yet hopeful attitude I want the next generation of software engineers to understand. Computer science has just begun.

Additional References

Clancey, W. J. (1995a). AI: Inventing a New Kind of Machine. *ACM Computing Surveys, 27*(3), 320-322.

Clancey, W. J. (1995b). Practice cannot be reduced to theory: Knowledge, representations, and change in the workplace. In S. Bagnara & C. Zuccermaglio & S. Stucky (Eds.), *Organizational Learning and Technological Change* (pp. 16-46). Berlin: Springer. (Papers from the NATO Workshop, Siena, Italy, September 22-26, 1992.)

Clancey, W. J. (1997a). *Situated Cognition: On Human Knowledge and Computer Representations.* New York: Cambridge University Press.

Clancey, W. J. (1997b). The conceptual nature of knowledge, situations, and activity. In P. Feltovich & R. Hoffman & K. Ford (Eds.), *Human and Machine Expertise in Context* (pp. 247–291). Menlo Park, CA: The AAAI Press.

Clancey, W. J. (1999a). *Conceptual Coordination: How the Mind Orders Experience in Time.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Clancey, W. J. (in press). Human exploration ethnography: The Haughton-Mars Project 1998-1999. *Proceedings of the Second Annual Meeting of the Mars Society*. Boulder, CO.

Dewey, J. ([1896] 1981). The reflex arc concept in psychology. *Psychological Review,* III (July), 357-370. (Reprinted in J.J. McDermott (ed.), The Philosophy of John Dewey, Chicago: University of Chicago Press, pp. 136-148.)

Dobson, J. (1996). Desert Island Column. *Automated Software Engineering Journal*, vol. 3, no. 1/2 June.

Lopez, B. (1986). *Arctic Dreams: Imagination and Desire in a Northern Landscape*. New York: Bantam Books.

M.A.R.S. (1999). The Mars Arctic Research Station. http://www.marssociety.org.

Ryan, K. (1996). Desert Island Column. *Automated Software Engineering Journal*, vol. 3, no. 3/4, August, pps. 391-393.

Zubrin, R. and Wagner, R. (1996). *The Case for Mars. The Plan to Settle the Red Planet and Why We Must*. New York: Free Press.